2012

Experimenting with Anomaly Detection by Mining Large-Scale Information Networks

A. Taleb-Bendiab
Edith Cowan University

DOI: 10.4225/75/57b55634cd8d6
Originally published in the Proceedings of the 10th Australian Information Security Management Conference, Novotel Langley Hotel, Perth, Western Australia, 3rd-5th December, 2012

EXPERIMENTING WITH ANOMALY DETECTION BY MINING LARGE-SCALE INFORMATION NETWORKS

A.Taleb-Bendiab
School of Computer and Security Science, Edith Cowan University
Perth, Western Australia
a.taleb-bendiab@ecu.edu.au

Abstract

Social networks have formed the basis of many studies into large networks analysis. Whilst much is already known regarding efficient algorithms for large networks analysis, data mining, knowledge diffusion, anomaly detection, viral marketing, to mention. More recent research is focussing on new classes of efficient approximate algorithms that can scale to billion nodes and edges. To this end, this paper presents an extension of an algorithm developed originally to analyse large scale-free autonomic networks called the Global Observer Model. In this paper, the algorithm is studied in the context of monitoring large-scale information networks. Hence, taking into account the size of such networks, the proposed algorithm starts by partitioning the graph using structural network metrics. This is followed by a calculation of the graph nodes' metrics, which are used in the selection from the original graph a subset of nodes to be monitored. The paper is organised as follows: it will outline the problem definition and algorithm, then will proceed to a brief description of an event and signature based model used to instrument monitored nodes. Finally, the paper will conclude with an evaluation using an infection detection scenario, which will be followed by a general discussion and proposed further work.

Keywords
Graph analysis and mining, social networks, graph clustering.

INTRODUCTION

The last decades have seen a flurry of interest and activities related to research into information networks analysis, be it, to study methods for patterns discovery, event correlation, and/or knowledge discovery from large-scale networks, Kwak et al. (2010). Many real-world applications have been studied ranging from: (i) social networks analysis, Richardson and Domingos (2002), Barabasi et al. (2000) including: information propagation and cascades, Bollobas and Riordan (2003), Kawachi et al. (2008), influence of nodes characterisation, Kleinberg (1999), Brin and Page (1999) and community structure detection, Domingos and Richardson (2001), Holder et al. (1994), Newman (2004), Krebs (2001); (ii) anomaly detection, Montanaria and Saberib (2010), Mogul (2006), to mention but a few.

While much is already known regarding social networks, and the application of high-performance computing for large-scale networks analysis, yet with the "big-data" phenomenon a body of recent research is now focusing on new classes of efficient yet approximate algorithms which can cope with the billions-scale type of networks such as Twitter (Kwak et al. 2010). As argued by many, (Leskovec et al., 2007; Yang and Leskovec. 2010; Kang et al., 2011) the analysis of such extremely large networks are too slow to compute. For instance, the computation of a single metric such as the centralities measure is known to run in the order of $O(n^3)$ with n being the number of nodes in a graph, Brandes (2001), Newman (2005). Thus, the complete (or exhaustive) analysis of such graphs will be slow and may only be attempted on an off-line mode, Yang and Leskovec (2010).

However, recent results emerging from studies into social networks are uncovering various cases of self-similarity patterns and modularity of not only the networks' structure, users' behaviour, but also the data generated. These findings are providing a rich source of useful heuristics, which are used for instance: for data reduction and/or parallel execution of networks analysis algorithms, Barabasi et al. (2000), Jeong et al. (2001), Newman (2003), Yan and Han (2002).

Along this line of work, this paper presents recent results related to a study into algorithms for monitoring large-scale information networks to detect global events under incomplete information settings, Randles et al. (2010a, 2010b), Lamb et al. (2007, 2009). The paper is structured as follows: A general definition of the problem is followed by a description of the method and algorithm used for large-scale graph monitoring. Then the paper presents a laboratory-based evaluation of the algorithm, which uses the "infection diffusion" problem to evaluate the algorithm. Finally, the paper concludes with a general discussion, concluding remarks and suggested further work.

PROBLEM DEFINITION

Taking into account the scalability concern stated above, in order to monitor the dynamics (such as information cascade, influence) of a given social network by mining its graph will require addressing a number of aspects including:

- Graph partitioning: this requires the study of generic and domain specific heuristics to facilitate the graph partitioning problem. For instance, networks self-similarity and modularity properties, which as shown by Leskovec et al. (2012) can reduce the amount of data been analysed, where 2.5 million blogs collected were reduced to 45000 blogs relevant to their experiment. Also, Randles et al. (2008) showed in their study that the graph size can be reduced by 17% while still providing 90% knowledge of the network state.
- Accuracy: this requires the study of methods to estimating the accuracy and/or loss of information, which is vital in assessing the trade-off of computational cost versus information loss of a considered algorithm.
- Computing the minimal monitoring subgraph: this requires the study of decision supports for the selection of the "best" node to monitor, Montanaria and Saberib (2010), Lamb et al. (2009).

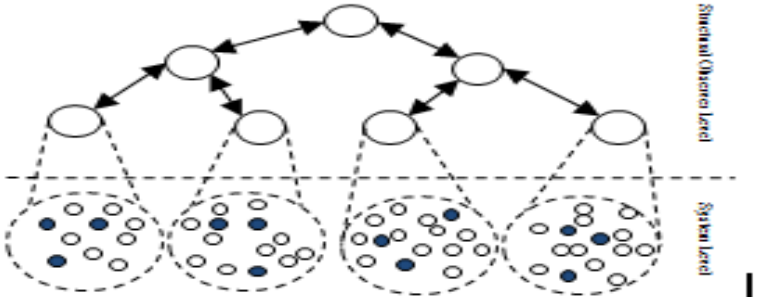

Figure 1: Hierarchical organisation of structural observers.

METHOD

We represent a network under consideration as an undirected graph $G = (V, E)$ with V a set of vertices and E a set of edges. We define a cluster C_1 in graph G as $C_i = (V_i, E_i)$ where V_i is a subset of V. Thus $\{V_1, \ldots, V_k\} = V$,

and $\{E_1, \ldots, E_k\} \subseteq E$.

We define $O = \{O_1, \ldots, O_m\}$ as the set of observer nodes, which are mapped by a function M to a cluster's representative nodes (Fig. 1). The selection of the representative nodes per cluster is sometimes selected as the centroid of the subgraph, Wu et al. (2004). Similarly to Leskovec et al. (2007) and Kang et al. (2011) we calculate the cost function c_{vi} of a node vi as:

$$c_{vi} = \sum_{i=1}^{n} a_i c_i$$

which combines the node structural metric in the graph and the node's decision cost. This is detailed below.

As illustrated by Figure 1, the observer graph forms a virtual fully connected graph. Other approaches inspired by the BIRCH model, Zhang et al. (1996) use a trees data structure to represent such hierarchical structure of clusters to representative nodes.

```
Algorithm: Observer Graph Computation
computeObsGraph(Graph g) {
        List<Communities> communities = graphPartition(g);
        for (Community c : communities) {
             computeNodeMetrics(c);
             observeNodeSet = communityToObserve(c);
             observeNodeMap.add(observeNodeSet);
```

```
        }
    }
```

```
    communityToObserve(Communities c) {
        Collection<Node> nodes = c.getNodeSet();
        List<Community, Node, Cost> rankedMSet;
        for (Node n : nodes) {
            cost = n.getNodeCost;
            rankedMSet.add(c, n, cost);
        }
        Comparator comparator = new MyComparator();
        Collections.sort((List) rankedMSet, comparator);
        observeNodeSet = rankedMSet.select(criteria);
        return observeNodeSet;
    }
```

Figure 2: Java-like pseudocode of aspects of the developed algorithm.

ALGORITHM

As illustrated in Figure 2, the algorithm for monitoring a graph *G* is performed as follows:

- Partitioning the graph using structural network metrics. Similarly to, Wu et al. (2004) we use betweeness centrality to perform the partitioning of the graph (Fig. 3), which follows Newman's (2004) method for community detection. For this we use Brandes (2001) algorithm to computer nodes betweeness centrality, which runs on a weighted graph at O(nm + n^2 log n) time complexity.

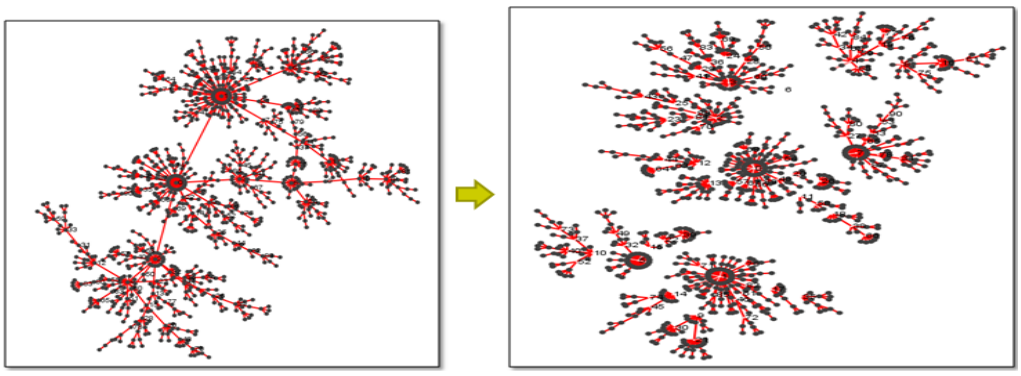

Figure 3. Visualisation of the graph partitioning process.

- To compute the subgraph of nodes to be monitored – the placement of monitoring points – is a multi-criteria optimisation problem to be computed on the whole graph is an NP hard problem. Thus, the algorithm performs this in a number of steps: (i) for each community (or partition), it calculates its nodes' cost function. The latter is computed as a node structural metric using betweeness measure – others measure are also possible such as node's degree, PageRank, Hits. (ii) nodes' cost function is used to select a defined proportion of nodes with maximum/minimum cost, which represents the observer (or monitored) nodes set for each graph partition.
- Instrumenting monitored nodes using the Global Observer Model describer below.

ARCHITECTURAL MODEL OF THE GLOBAL OBSERVER

In order to meet some of the engineering requirements to automating the monitoring of large-networks, a *Global Observer Model* was proposed and first described in Lamb et al. (2007, 2009). This model is a "distributed" implementation of the standard observer software pattern, Rosenblum and Wolf (1997), Beck et al. (2000). The model uses a signature-based condition event action to detect and dispatch events. As illustrated in Figure 4 the model provides a number of features:

- Modelled Elements: these provide the description of the structure to be assessed, such as the collection of set of monitored nodes and associated event listeners.
- Model Change Events: these provide the descriptions for the invalidation mechanism. In other words the event handler.
- Signature Template: implementations are responsible for: specifying the match criteria, the model to assess, and for assessing the match criteria and providing a result.
- Invalidation Handler: the invalidation handler should provide functionality (to be used by an interested party, such as an observer) to determine when a signature should be reassessed.

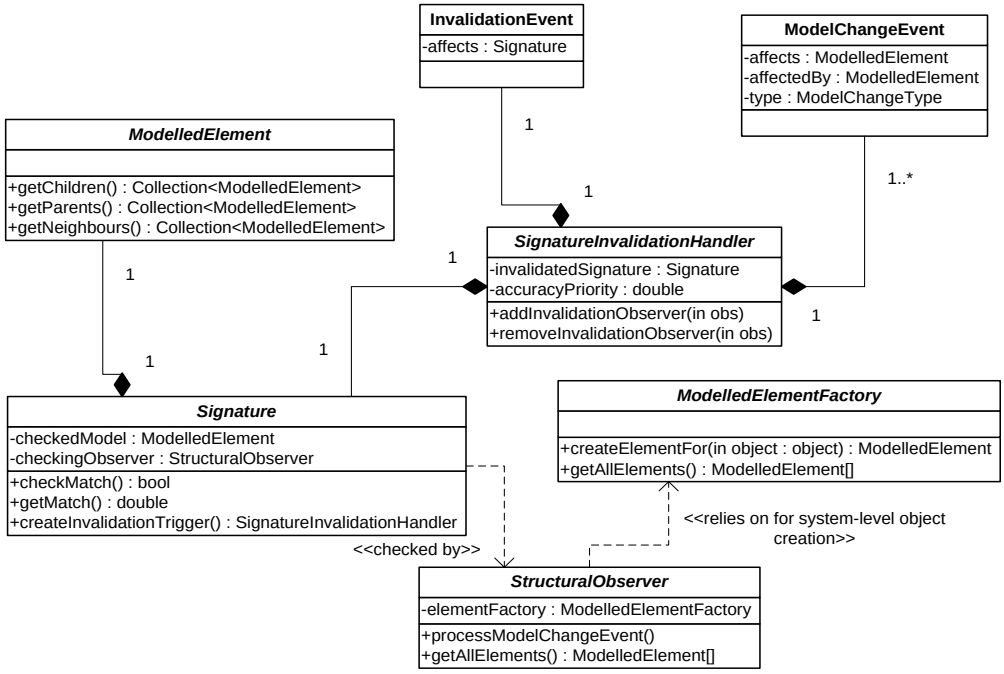

Figure 4: Signature and invalidation handler - significant classes, relationships and methods

In this work, we use this event-based model to support the runtime anomaly detection. Although, other methods using time series analysis of sampled measures (metrics) are also usable.

EXPERIMENT

In order to evaluate the algorithm, we used the "infection" detection scenario to experiment with the different monitoring strategy. Thus the evaluation focusses on the system's ability to keep an observed system "protected" against an introduced infection. The diffusion of infection is used here to measure the success or otherwise of infection protection (or monitoring) employed.

As shown in Figure 5, immunised nodes are shown in green. Infected nodes are shown as red, while "normal" nodes are blue. The size of the node indicates the number of connections to other nodes (the degree).

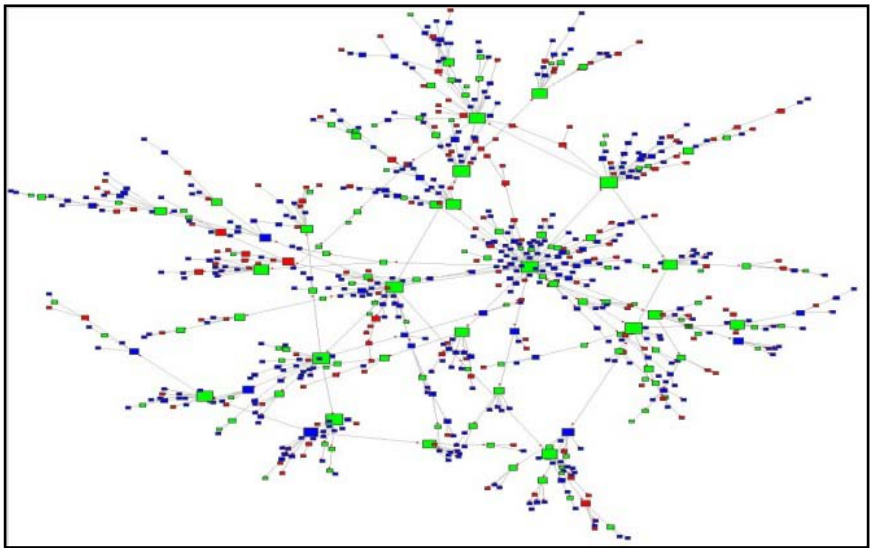

Figure 5 Infection simulation network screenshot

An infection is introduced to a randomly selected set of nodes of a graph G. The infection then propagates from one node to another with a probability *p*. The validation experiment compares the infection propagation with: (i) no monitors deployed (referred to as *none strategy*), (ii) randomly selected monitored nodes (or *random strategy*), (iii) used global observer model to select monitored nodes (or *intelligent strategy).*

In addition, the implementation of the global observer model was assessed based on two main criteria:

- The cost of the selected observation strategy: we simplify this to count how many nodes needed observation under each monitoring strategy for the same simulation settings including: graph size, propagation model, and experiment duration.
- The effectiveness of protection: we calculate how many nodes were infected at the end of a given simulation run.

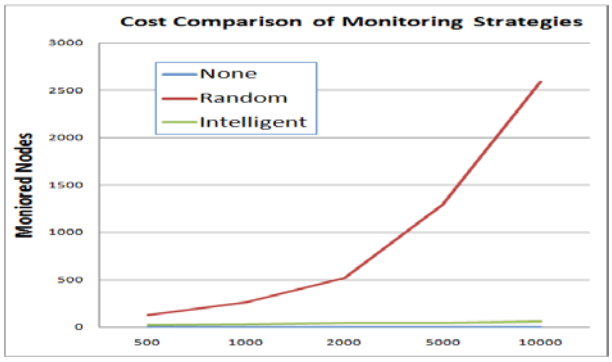

Figure 6: Cost of monitoring strategies (Network Size vs. Nodes Observed)

DISCUSSION

As illustrated in Figure 6, the cost of observation – in terms of the number of deployed units at the system-level – the *None Strategy* is the lowest cost, while *Random Strategy* increases with the size of the system. However, the *Intelligent Strategy*, while increasing with the overall size of the system, is a very low-cost option. This suggests that the intelligent strategy is potentially selecting the observation targets efficiently.

In addition, the Figure 7 indicates the number of nodes infected after the infection attempt simulation; the plot for *None Strategy* indicates an open control – the effective spread of the infection algorithm without any observer protection.

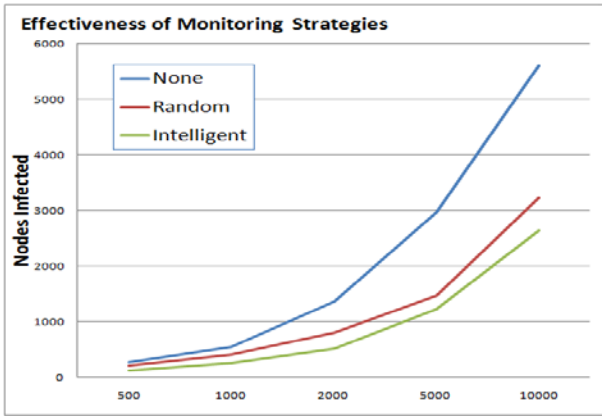

Figure 7: Effectiveness of monitoring strategies (abscissa: Network Size)

CONCLUSIONS

Recent research in information networks and social networks analysis is focusing on new classes of efficient yet approximate algorithms which can cope with the billions-scale type of networks such as Twitter, Kwak (2010). As argued by many, Leskovec et al. (2012), Yang and Leskovec (2010), and Kang et al. (2011) the analysis of such extremely large networks are too slow to compute. For instance, the computation of a single metric such as the centralities measure is known to run in the order of $O(n^3)$ with n being the number of nodes in a graph, Brandes (2001), Newman (2005). Thus, the complete (or exhaustive) analysis of such graphs will be unrealistic and may only be attempted on an off-line mode, Yang and Leskovec (2010). New results emerging from studies into social networks are uncovering various cases of self-similarity patterns and modularity of not only the networks' structure, users' behaviour, but also the data generated. These are providing a rich source of useful insight and heuristics, which are used in selective data reduction and/or parallel execution of networks analysis algorithms, Barabasi et al. (2000), Jeong et al. 2001), Newman (2003), Yan and Han (2002).

However, careful design and analysis is required to exploit and apply such heuristics. Along this line, this paper presented results related to the development of an approximate algorithm for monitoring large-scale stochastic information networks. This study focussed specifically on methods for detecting global events under incomplete information settings, Randles et al. (2010a, 2010b), Lamb et al. (2007, 2009).

The paper presented a general definition of the problem and method used. This was followed by a description of the proposed monitoring algorithm, which is used to compute an optimum set of nodes to monitor – referred to here as the observer subgraph. The algorithm performs this task in the following sequence:

- Partitioning the graph using structural network metrics: this is performed following the Newman's method for community detection, Newman (2005). In other words, through an iterative process until communities emerge (Fig. 3), it calculates nodes' betweeness centrality measure using an implementation of Brandes (2009) algorithm. Then removes edges with the highest betweeness centrality measure.
- For each community (or partition) the algorithm calculates its nodes' cost function. The latter is computed as a node structural metric using betweeness measure – others measure such as node's degree, PageRank, Hits are also possible. Then nodes' cost measure is used to select a defined proportion of nodes with maximum/minimum cost to represent the observer (or monitored) nodes set for each graph partition.
- Each monitored nodes is instrumented following the Global Observer Model architecture.

A laboratory-based evaluation of the algorithm using synthetic graphs and an "infection diffusion" scenario shown that the Global Observer Model, Lamb et al. (2007) can reduce the size of monitored subgraph by 17% of the original graph while still provides 90% knowledge of the graph state, Lamb et al. (2009). In addition, as expected the algorithm has shown an improved protection via the use of the global observer layer.

Whilst the laboratory experiments have shown positive results, more tests are required. In particular, a further evaluation of the algorithm is underway to apply the algorithm to outlier detection using real dataset of the autonomous system (graph of the Internet) from the Stanford Large Network Dataset Collection, SNAP (2012). In addition, further work is required to investigate (i) the distributed implementation of the algorithm and its performance analysis, (ii) interplay of time-series analysis and event-based model for anomaly detection.

ACKNOWLEDGEMENTS

The contribution of the following people was essential for the performance of this work, including:

- Dr. David Lamb and Dr. Martin Randles, School of Computing and Mathematical Sciences, Liverpool John Moores University, who with the author developed and implemented the original Global Observer Model and associated algorithms.
- Mr. Peter Hannay, School of Computer and Security Science, ECU, for providing MySQL Twitter data set. A
- Mr. Dan Pitic and Mr. Zhiji GU, students at School of Computer and Security Science, ECU for assisting with the implementation of a port of the MySQL dataset to Neo4J graph-database, and Twitter streaming for data collection. Unfortunately, due to lack of space this aspect of the work has not been included.

REFERENCES

Barabasi, A.L., Albert, R. & Jeong, H. (2000). Scale-free characteristics of random networks: the topology of the world-wide web, *Physical and Statistical Mechanics and its Applications, 281*, 69-77.

Barabasi, A.L., Albert, R., Jeong, H. & Bianconi, B. (2000). Power-law distribution of the world wide web. *Science, 287*.

Beck, K. Fowler, M. & Kohnke, J. (2000). *Planning Extreme Programming*: Addison-Wesley.

Bollobas, B. & Riordan, O. (2003). Robustness and Vulnerability of Scale-Free Random Graphs," *Internet Mathematics, 1*, 1-35.

Brandes, U. (2001). A Faster Algorithm for Betweenness Centrality. *Journal of Mathematical Sociology, 2001, 25*(2), 163-177.

Brin, S. & Page, L. (1998). The anatomy of a large-scale hypertextualWeb search engine. In Proceedings of the *SeventhInternational Conference on World Wide Web 7*, pp.107–117. Elsevier Science Publishers B. V.

Domingos, P. & Richardson, M. (2001). Mining the network value of customers. In Proceedings of the Seventh *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 57–66. ACM Press.

Holder, L., Cook, D. & Djoko, S. (1994). Substructure discovery in the subdue system. In Proceedings of the *Workshop on Knowledge Discovery in Databases*, pp.169–180.

Jeong, H., S. P. Mason, A. L. Barabasi, and Oltvai, Z. (2001). Lethality and centrality in protein networks. *Nature, 411*, 41–42.

Kang, U., Papadimitriou, S., Sun, J. & Tong, H. (2011). Centralities in Large Networks: Algorithms and Observations. SDM 2011: 119-130.

Kawachi, Y., Murata, K., Yoshii, S. & Kakazu, Y. (2008). The structural phase transition among fixed cardinal networks," *Complexity International, 12*, msid43.

Kleinberg, J. (1999). Authoritative sources in a hyperlinked environment. *J. ACM, 46*(5):604–632.

Krebs. V. (2001) .Mapping networks of terrorist cells. *Connections, 24*.

Kwak, H. Lee, C., Park, H., Gwahangno, S. M. & Daejeon, Y. (2010). What is Twitter, a Social Network or a News Media? WWW 2010, ACM 978-1-60558-799-8/10/04.

Lamb, D., Randles, M. & Taleb-Bendiab, A. (2007). Software Engineering Concerns in Observing Networks of Autonomic Systems. *System and Information Sciences Notes Journal, 2*,101-104.

Lamb, D., Randles, M. & Taleb-Bendiab, A. (2009). Monitoring Autonomic Network through Signatures of Emergence, *IEEE Conference and Workshops on Engineering of Autonomic and Autonomous Systems*, EASe 2009, pp: 56-65.

Leskovec, J., Krause, A., Guestrin, C., Faloutsos, C. VanBriesen, J. & Glance, N. (2007). Cost-effective Outbreak Detection in Networks, Internal report of Computer Science Department, Carnegie Mellon University, paper 528. Retrieved from http://repository.cmu.edu/compsci/528

Montanaria, A. & Saberib, A. (2010). The spread of innovations in social networks, proceedings of the *National Academy of Sciences of the United States of America (PNAS), 107* (47), pp. 20196-20201.

Mogul, J. (2006). Emergent (mis)behavior vs. complex software systems," *ACM SIGOPS Operating Systems Review*, *40*, 293 - 304.

Newman, M. (2003). The structure and function of complex networks. *SIAM Review*, *45*, 167–256.

Newman, M. (2004). Detecting community structure in networks, In Eur. Phys. J. B.

Newman, M. (2005). A measure of betweenness centrality based on random walks, *Social Networks*.

Randles, M., Abu-Rahmeh, O., Johnson, P. & Taleb-Bendiab, A. (2010). Biased random walks on resource network graphs for load balancing, Journal: *The Journal of Supercomputing - TJS , 53*(1), 138-162.

Randles, M., Lamb, D., Odat, E. & Taleb-Bendiab, A. (2010). A Distributed Redundancy and Robustness Analysis Method for Networks of Autonomic Systems, *The Journal of Computing and System Sciences*, Elsevier.

Richardson, M. & Domingos, P. (2002). Mining knowledge-sharing sites for viral marketing. In Proceedings of the *Eighth ACM, SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp.61–70. ACM Press.

Randles, M. Lamb, D. & Taleb-Bendiab, A. (2008). Engineering Autonomic Systems Self-Organisation, in *EASe'08: Proceedings of the 5th IEEE International Workshop on Engineering of Autonomic and Autonomous Systems*.

Rosenblum D. S. & Wolf, A. (1997). A Design Framework for Internet-scale Event Observation and Notification, *SIGSOFT Software Engineering Notes*, *22*.

SNAP. (2012). Stanford Large Network Dataset Collection, http://snap.stanford.edu/data/Yan, X. & Han, J. (2002). gSpan: Graph-based substructure pattern mining. In Proc. *2002 Int. Conf. on Data Mining (ICDM'02)*.

Wu, A.Y. Garland, M. & Han, J. (2004). Mining Scale-free Networks using Geodesic Clustering, Proceedings of the *KDD'04*.

Yang, Y., & Leskovec, J. (2010). Modeling Information Diffusion in Implicit Networks, Proceedings of the *Data Mining (ICDM)*, ieeexplore.ieee.org

Zhang, T., Ramakrishnan, R. & Livny, M. (1996). BIRCH: an efficient data clustering method for very large databases. In *ACM SIGMOD Intl. Conf. on Management of Data*, pp.103–114, June.